(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 061 437 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
20.12.2000 Bulletin 2000/51

(51) Int Cl.[7]: **G06F 9/318**

(21) Application number: **99830375.4**

(22) Date of filing: **16.06.1999**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(71) Applicant: **STMicroelectronics S.r.l.**
20041 Agrate Brianza (Milano) (IT)

(72) Inventors:
- **Arcidiacono, Liliana**
  95030 Tremestieri Etneo (CT) (IT)
- **Matranga, Vincenzo**
  90143 Palermo (IT)

(74) Representative: **Botti, Mario**
Botti & Ferrari S.r.l.
Via Locatelli, 5
20124 Milano (IT)

(54) Improved control unit for electronic microcontrollers or microprocessors

(57) The invention relates to a control unit for electronic microcontrollers or microprocessors, and a method of fabricating said unit. The unit is of the type which comprises a finite state machine having at least one combinational network; the finite state machine is constructed of a plurality of control sub-units (2), each corresponding to one combinational logic network, each unit in said plurality of control sub-units (2) being independently connected to an arbitration block (3) to provide information about a possible future state and receive a present state command.

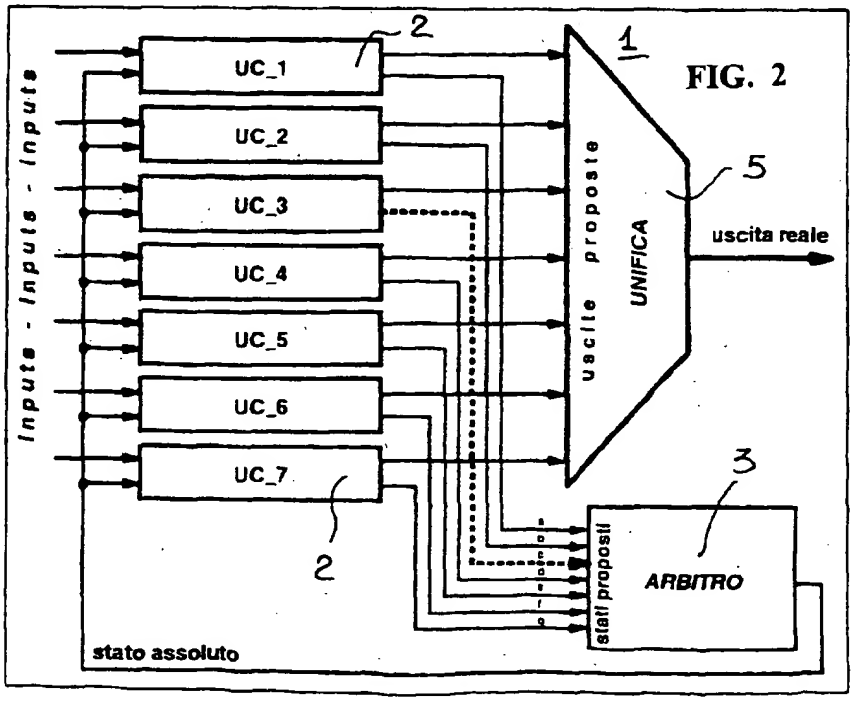


FIG. 2

EP 1 061 437 A1

Description

Field of the Invention

[0001] This invention relates to an improved type of control unit for electronic microcontrollers or microprocessors.

[0002] The invention specifically relates to a unit as above which comprises a finite state machine having at least one combinational logic network.

[0003] The invention also relates to a method of fabricating a control unit for electronic microcontrollers or microprocessors.

[0004] As is well known in this specific technical field, all microcontrollers or microprocessors are bound to include: a resource-managing central block provided inside a so-called device "core", and a set of peripheral blocks connected to the central block.

[0005] The managing central block is commonly referred to as the CU (Control Unit). A CU unit oversees all the operative phases of the microcontroller or microprocessor, and clocks the carrying out of some internal operations.

[0006] In programmable devices, the CU would perform, in principle, the exemplary operations summarized herein below.

1. Point the memory which contains a program row to be executed.
2. Decode the instruction thereof.
3. Carry out the operations included in the decoded instruction.
4. Point the next program row.

Prior Art

[0007] The complexity of devices based on a microcontroller or microprocessor is increasing, and this reflects in the device having to carry out a larger number of different operations, among which are the following:

- loading data to/from the device outside world;
- executing logic-arithmetic functions;
- conditional or unconditional skip operations, and subroutine calls;
- controlling interrupt signals;
- establishing special flows of operations, as is the case with microcontrollers or microprocessors designed for special applications, such as dedicated microcontrollers for fuzzy logics computations.

[0008] Thus, the larger the number of instructions, and hence the higher the computational capacity of the device to be designed, the greater becomes of necessity the hardware complexity of the UC unit. The latter is to recognize the instructions and deliver the appropriate sequence of control signals for the device to operate correctly, and is to retain its capability to operate in an exclusive manner where special asynchronous interrupt signals are required.

[0009] Today's designs for microcontrollers or microprocessors, as well as for the so-called ASIC (Application Specific Integrated Circuit) devices follow descriptive criteria.

[0010] In other words the circuit, instead of being designed by connecting circuit-wise such standard sub-units as logic gates to perform certain logic operations (AND, OR, NOT, etc.) in order to provide complex functions of relational operators, is defined by means of a descriptive language based on peculiar syntax and semantics.

[0011] One of these description languages is known by the acronym VHDL (VHSIC Hardware Description Language, where VHSIC stands for Very High Speed Integrated Circuit). This language provides the designer with a powerful tool for:

- translating the notional idea into a functional description;
- simulating the block thus described;

converting the behavior description to an RTL (Register Transfer Level) scheme.

[0012] The specifications for a block operation are formalized using the VHDL language in a listing. The integrated circuit design environments, known in the trade as Cadence, Synopsys, etc., allow of the creation of symbols, their behavior description, and ensuing simulation. The latter is aimed at investigating the correct operation of the block just described.

[0013] A proper behavior description can later be translated to an RTL, using compilers which effect the circuit synthesis to obtain a schematic consisting of a set of elementary logic gates.

[0014] The exemplary designing method outlined hereinabove is currently widely used for progressing from the notional idea to a hard implementation. However, it cannot solve the problem of providing highly complex functions.

[0015] In fact, the more are the functions that the circuit is to perform, the greater becomes the complexity and length of the listing in the VHDL code.

[0016] In a reductive way, it could be thought that this is merely a problem of listing size. Row-wise complex and bulky listings are indeed more difficult to set up, involve time-consuming checking procedures for servicing and modifying parts of it, and result in execution times which are directly proportional to their length in the respect of simulations and circuit "syntheses".

[0017] CU units provide a fair example of how the functional characteristics of a block turn into VHDL listings of a size and complexity which exceeds that of any other blocks within a microcontroller or microprocessor.

[0018] The overall construction of an FSM (Finite State Machine) basically comprises two combinational logic networks relating to the future state of the machine and the current outputs, as well as a sequential portion made up of a set of flip-flop memory cells for storing up given state vectors.

[0019] The inputs, and the vector representing the "present state" of the state machine, generate a "future state" vector that constitutes the data to be stored in the state memory cells. The outputs from these flip-flop cells retain the value of the present state, which is fed back to the combinational network for determining the value of the future state and is simultaneously supplied to a section of the current outputs, this section issuing new outputs of its own dependent on the present state, and occasionally inputs as well, as shown schematically in the accompanying Figure 1.

[0020] Considering the complex circuit and functional aspects of a control unit, it can be appreciated that this prior approach using a finite state machine does involve highly complex VHDL listings, especially in the respect of the combinational networks.

[0021] The underlying technical problem of this invention is to provide a novel type of control unit, for microcontrollers or microprocessors of all description, which has such structural and functional features as to simplify the unit designing stage to the utmost degree and overcome the limitations of prior art solutions.

Summary of the Invention

[0022] The concept behind this invention is one of having the control unit split into a plurality of sub-units or assembly sub-units, each to perform a specific function, so that they can be fabricated independently and tested separately.

[0023] In essence, the concept on which the invention stands provides for the control unit to be suitably resolved into classes and/or phases of operation whereby the complexity of each modular block in the whole unit can be effectively reduced.

[0024] Based on this concept, the technical problem is solved by a control unit as previously indicated and defined in the characterizing part of Claim 1.

[0025] The features and advantages of a control unit according to the invention can be more clearly understood from the following description of an embodiment thereof, given by way of non-limitative example with reference to the accompanying drawings.

Brief Description of the Drawings

[0026] In the drawings:

Figure 1 shows schematically a conventional control unit incorporating a finite state machine;

Figure 2 shows schematically a control unit according to this invention;

Figure 3 shows schematically an embodiment of the control unit of Figure 2;

Figures 3A and 5 are respective schematic views illustrating the unit of Figure 2 in different conditions of its operation;

Figures 4, 6 and 7 are respective schematic views of embodiments of the control unit according to the invention.

Detailed Description

[0027] With reference to the drawing views, and particularly to the embodiment shown in Figure 2, a control unit according to this invention and useful with electronic microcontrollers and/or microprocessors is shown generally at 1 in diagramatic form.

[0028] The unit 1 is, particularly but not exclusively, intended for a fuzzy microcontroller manufactured by the Applicant as model ST52E402. This specification will cover that specific field of application for convenience of illustration only, it being understood that the control unit 1 of this invention can be used with any types of electronic microcontrollers or microprocessors.

[0029] The unit 1 is constructed of a number of control sub-units 2, denoted by the references UC_1,...,UC_7, each adapted to be defined, fabricated or designed separately; an arbitration block 3 is arranged to oversee and control the operation of the several sub-units 2, as well as those operations which enable operation even of a single one of the sub-units comprising the control unit 1.

[0030] Advantageously in this invention, to prevent the arbitration block 3, as the decision-maker on the operations to be undertaken, from growing to be a complex circuit in itself, a particular management is provided for the various sub-units 2.

[0031] Each control sub-unit 2 is independently connected to the block 3, and supplies a predetermined value representing a state of operation to the arbitration block 3. This state may be an accepted one by the block 3 or be a neutral state.

[0032] The arbitration block 3 is to maintain the one possible state of operation.

[0033] Accordingly, the arbitration block acts to relay signals. When the structure of this invention is regarded as the equal of a conventional finite state machine, it is as if each sub-unit 2 corresponded to a combinational logic network and the unit 1 comprised a plurality of combinational networks, one for each class of operations or instructions. The machine flow provides a set of states for each of the combinational networks. Each sub-unit 2 proposes, according to an instruction to be executed, a possible state of its own, and at any given time, only one of them will have its state coincident with one of those of the state vectors, with all the others in a so-called neutral state, designated NEUTRO in the drawings.

[0034] The proposed states are collected by the arbitration block 3, which will decide on which of the sub-units 2 is to keep proposing the future state or stay in the neutral state. In addition, this block 3 contains a sequential network 4 storing the state vector.

[0035] Referring to Figure 2, each sub-unit 2 represents a portion of the control unit which is only activated when the absolute present state of the unit 1 is among those accepted by the sub-unit. In this case, the sub-unit will independently produce all the control signals required for correctly executing relevant instructions or operations, and supply a future state of the whole unit 1.

[0036] If, on the contrary, the individual sub-unit 2 is not activated, it will propose a so-called NEUTRO state.

[0037] All of the states proposed by the individual sub-units 2 forming the control unit 1 are connected to the arbitration block 3 which will allot the present state while taking into account the future state proposed by each sub-unit.

[0038] The respective outputs of each control sub-unit 2 are also connected to a current output collecting block 5, as shown in Figure 5, which will select the single output having a value other than NEUTRO.

[0039] For example, referring to Figure 2, during the execution of an instruction which is included in the class of instructions relating to the third sub-unit UC_3 represented, the operations will be managed only by that control sub-unit UC_3. Thus, all the other sub-units 2 will supply the proposed state: a = b = d = e = f = g = NEUTRO, and the single active signal to determine the present state, and hence the current output, will be the signal "c", i.e. the signal from the control sub-unit UC_3 involved.

[0040] The absolute state of the control unit 1 will match that determined by the sub-unit UC_3 until all the states relating to the execution of the standing instruction are completed. Also, this control sub-unit UC_3 will eventually propose a state that does not belong to it, and then enter the state NEUTRO, so that the absolute state will be originated by the sub-unit 2, which takes over in the control. All of the signals required for correct operation of the whole control unit 1 are, therefore, generated by the sub-unit that takes over in the management; any other sub-units generating the same signals will supply non-influential neutral values inside the output collector block 5. In fact, the block 5 collects all the signal of common logic value, which will be ORed or ANDed together according to whether the active value is a logic 1 or 0.

[0041] Shown in Figure 3 is a schematic example of how the state signals are processed by the unit 1 as a whole. It is assumed that the control unit 1 can be broken down into three basic blocks, denoted UNO, DUE and TRE, only; each block proposes a new future state for the whole control unit 1 by means of signals dice-UNO, dice-DUE and dice-TRE. The arbitration block 3 processes these three signals and allots the correct absolute state upon receiving a

successive active edge of a synchronization signal, i.e. a so-called CLOCK, which represents the timing of the unit 1.

[0042] The accepted states by the control unit 1 (NEUTRO, IDLE, PRIMO, SECONDO, TERZO, QUARTO, QUINTO, SESTO, SETTIMO) can be defined by means of a defining block 6 designated "package". In the example of Figure 3, the package block 6 comprises a list of the sub-units 2 and is compiled, allotted to a library stati_UC, and entered in each listing written in the VHDL code.

[0043] The string type, i.e. a variable formed of a set of alphanumeric characters, is defined within the package 6; in this way, it becomes unnecessary to set a binary code for each state beforehand. A device of a so-called Design Analyzer SYNOPSIS to allot directly the relating code.

[0044] Thus, splitting the unit 1 into individual control sub-units 2 allows:

the designing job to be distributed among several people working independently of one another;

slimmer listings to be provided, and easier functional checking of the same;

a modular structure to be provided; each sub-unit can be regarded as an internal sub-unit of the unit 1. Sub-units such as these can be added or removed, and can be individually modified without involving the whole control unit 1.

[0045] It matters to point out that the proposed modular structure is not addressed to provide a capability of reducing the complexity of the listings and/or the algorithms which describe the control unit at the behavior level, the innovative aspect of this invention residing in the functional independence of each sub-unit 2.

[0046] The sub-units 2, being suitably divided among classes of instructions or operations, can be connected in or out independently, according to the user's requirements and the design specifications, without altering the architecture of the control unit 1.

[0047] The control unit 1 can be formed, according to the invention, by just assembling together such basic sub-units 2 as will provide the required performance, thereby optimizing the overall silicon area required by the microcontroller.

[0048] Alternatively, where fabrication time is a prevailing consideration over area occupation, the architecture of the control unit can be so laid out as to comprise the largest number of sub-units 2. Depending on the type of the application for which the machine is intended, a mechanism of activation/de-activation of the single sub-unit on instruction can be introduced.

[0049] For example, referring now to Figure 3A, the sub-unit UC_3 can generate an enable signal upon dedicated instruction which allows the unit UC_2 to be activated when input thereto.

[0050] The sub-unit UC_2, when activated, will evolve according to the state sequence set therefor each time that an instruction included among those contained in the class processed by the sub-unit is invoked. Conversely, when deactivated, the sub-unit UC_2 will propose at its output the so-called state NEUTRO, giving control over to another sub-unit.

[0051] This type of disabling obtained by means of a signal generated during operation, and therefore by the program, can also be provided by wiring the de-activation signal in the structure 1, where not all the class of instructions associated with the sub-unit is expected to be useful.

[0052] The control unit 1 of this invention offers several advantages in terms of the operations to be carried out in order to modify the unit when the latter is used within a class of products belonging to the same family.

[0053] The modifications are easily applied by virtue of the modular character of the control unit 1 lending itself for the addition or cancellation of one or more instructions and/or operations intended for execution.

[0054] In view of that the structure 1 has been used to develop an ST52E4XX family of microcontrollers/microprocessors by the Applicant, reference will be made to this family hereinafter.

Cancelling Instructions

[0055] An instruction can be cancelled by just acting on the sub-unit 2 which is to execute the corresponding set of instructions.

[0056] It will be recalled that each sub-unit 2 is to execute a set of instructions.

[0057] Cancelling an instruction involves acting on a single sub-unit, without altering the connections to the other sub-units; this results in the VHDL encoded listing being modified which describes the sub-unit at the behavior level, with no need to act on the schematic of the control unit 1.

[0058] For example, assume that the set of logic-arithmetic instructions is to be reduced by removing the instruction MIRROR: it will be sufficient to modify the sub-unit UC_ARITH_M6_TEST shown in Figure 4 by removing the VHDL code row relating to the decoding (fetch) of the microcode word. Thus, it will suffice if the following rows are modified within the state ADD-AR1:

```
when ADD_AR1 =>
     if dok="1" then
          if word_1 = "0101" or NOT
             word_1 = "1001" or ASL
             word_1 = "1010" or ASR
             word_1 = "1100" or DEC
             word_1 = "1101" or INC
             word_1 = "1011" then MIRROR
               future state <= OPER;
else
               future state <= STABIL_1;
end if;
else
     future state <= current state;
end if;
```

with the following code segment

```
when ADD_AR =>
     if dok = "1" then
          if word_1 = "0101" or NOT
             word_1 = "1001" or ASL
             word_1 = "1010" or ASR
             word_1 = "1100" or DEC
             word_1 = "1101" thenINC
               future state <= OPER;
     else
               future state <= STABIL_1;
     end if;
     else
          future state <= current state;
     end if;
```

wherein the VHDL code row, relating to the decoding of the four LSBs of word-1 which identify the instruction MIRROR, is no longer present.

[0059] Likewise, it will be necessary to suitably cancel those code fragments which contain the decoding of word_1, or part thereof, relating to the allotment of a given value for the sub-unit outputs. Thus, the listing should be searched for the occurrence of the row type:
if word_1 = "00101011" than, or parts thereof:
(if word_1(3 downto) = "1011" then).

[0060] Since each sub-unit in the control unit 1 of an ST52E402 controller allows the whole microprocessor or microcontroller to be managed independently of the other sub-units, to complete the removal of an instruction from the whole set, it is advisable to cancel within each sub-unit (in the VHDL listing) the outputs generated by the removed instruction, which are present in the process relating to the output storing.

[0061] In particular, each sub-unit 2 is input the general synchronization (clock master) signal which causes all the outputs from the sub-units to be stored in flip-flops (preferably of the FD2 type, that is with asynchronous reset).

[0062] To minimize the requirement for silicon area resulting from the logic synthesis process, whenever an instruction is removed, those signals should be cancelled which are only generated by it in the process clocked by the occurrence of the synchronization signal and the presence of the reinitializing (reset) signal.

[0063] Likewise, where the removal of an instruction involves cancelling one or more states of the state machine FSM of the control unit 1, they should also be cancelled within the package "tipi_UC.vhd" 6, which package must be "re-compiled" before the next logic synthesis using the Design Compiler.

Cancelling a Class of Instructions

[0064] A whole class of instructions can be effected by merely eleminating the sub-unit from the schematic of the control unit 1.

[0065] Shown schematically in Figure 4 is an example, shown already in Figure 3, relating to a control unit 1 which is divided into three sub-units 2, from which sub-unit DUE has been eliminated.

[0066] In this case, it will be necessary to also cancel all the signal branchings which formed inputs to the removed sub-unit, and the outputs; among the latter is the output designated dice-DUE, which is input to the arbitration block 3, from where the code segment relating to the management of the signal dice-DUE proposed by the removed sub-unit must be cancelled.

[0067] In a similar way, it will be necessary to cancel inside the package 6 the list of all the states of the state machine in block DUE; in the example, the states TERZO, QUINTO and SESTO have been removed, with an attendant reduction of the sequential (flip-flop) network required to store the "state variable" of the finite state machine.

[0068] Referring to the diagram of the control unit 1 of the ST52E402 microcontroller shown in Figure 5, where the whole set of logic-arithmetic instructions E is to be removed, for example, it will be sufficient to cancel from the schematic the sub-unit UC_ARITH_M6_TEST, together with the input and output signals, as shown in Figure 4. In addition, all occurrences of signals from the removed sub-unit should be cancelled within the block UC_UNIFICA_M6_TEST 5.

[0069] In either instances of individual instructions or a whole class of instructions being cancelled, this may involve a reduction in the finite states defined within the package 6, which package should be compiled ahead of the logic synthesis.

Introducing Instructions

[0070] An instruction can be added to a class of instructions by merely including, in the VHDL listing that describes the sub-unit to which the subject class is referred, the decoding of the new instruction, along with the states required for carrying out the operations provided by that instruction.

[0071] Taking the example shown in Figure 3, the addition of an instruction could involve, for example, the inclusion of an additional decoding (elsif B = "1" then
dice-TRE <= OTTAVO;) and two new states (OTTAVO and NONO) to the sub-unit TRE; it also being necessary to add these states to the related package (elenco-UC), as shown in Figure 6.

Introducing a Class of Instructions

[0072] A whole class of instructions can be added by merely including a new sub-unit within the control unit 1.

[0073] A new sub-unit will generate a signal for proposal to the arbitration block 3 which conveys information about the newly introduced states. These states should be listed into the package 6 to be compiled before the logic synthesis.

[0074] For example, going back to the scheme proposed by Figure 3, assume that the sub-unit NEW is to be introduced, and that the process describing said sub-unit NEW at behavior level is responsive to two new signals (D and E), as shown in Figure 7.

[0075] The addition of new functions to the control unit 1 results in the introduction of the new sub-units with the indicated inputs and the output dice-NEW relating to the three new states: OTTAVO, NONO and DECIMO. These add to those present inside the package elenco_UC.

[0076] Any new output variables produced by the sub-unit NEW, if common with those generated by the other sub-units existing in the control unit 1, should be input to the block arranged to collect all the homolog outputs and render them unique.

[0077] In Figure 8, there is shown an example of a sub-unit, designated UC_NEW, being added into the schematic of the current control unit 1. The new set of states is proposed to the block UC_DECIDO_NEW, and simultaneously, all the output variables in common with the other sub-units are supplied to the block UC_UNIFICA_NEW 5, which will unify them.

[0078] Also in the instance of instructions or a class thereof being introduced, the package 6, created to contain the list of the states, is to be modified where the additional instructions are formed from a flow of states that were already present in the state vector.

[0079] The solution proposed by this invention has been used to implement a control unit in an ST52E402 microcontroller wherein each block had been formed by the "finite state machine" method.

[0080] For example, the control unit of ST52E402 was to manage:

11 loading instructions;

14 logic-arithmetic instructions;

9 skip and/or subroutine call instructions;

8 interrupt managing instructions;

further special instructions and pseudo-instructions for fuzzy computation.

[0081] Were this control unit formed as a single block, like in the prior art, this would have involved the formation of a single large listing difficult to check, modify or complete.

[0082] By contrast, forming the control unit as single sub-units 2, in accordance with the invention, permitted of:

distributing the designing work among several people;

obtaining slimmer listings easy to check function-wise;

providing a modular structure which can be added or subtracted modules.

Claims

1. An improved control unit (1) for electronic microcontrollers or microprocessors, of the type which comprises a finite state machine having at least one combinational network, characterized in that said finite state machine is constructed of a plurality of control sub-units (2), each corresponding to one combinational logic network, each unit in said plurality of control sub-units (2) being independently connected to an arbitration block (3) to provide information about a possible future state and receive a present state command.

2. A control unit according to Claim 1, characterized in that each sub-unit (2) is structurally and functionally independent of the other sub-units (2).

3. A control unit according to Claim 1, characterized in that each control sub-unit (2) supplies the arbitration block (3) with a predetermined value representing a predetermined state of operation or a neutral state.

4. A control unit according to Claim 1, characterized in that each sub-unit (2) proposes, based on the type of instruction to be executed, a possible state of its own to the arbitration block (3), and that only one in said plurality will be in an active state, all the others being in a so-called neutral state.

5. A method of fabricating a control unit for electronic microcontrollers or microprocessors, wherein said unit comprises a finite state machine having at least one combinational logic network, characterized by the finite state machine being constructed of a plurality of control sub-units (2), each corresponding to one combinational logic network, and each unit in said plurality of control sub-units (2) being independently connected to an arbitration block (3) for supplying information about a possible future state and receiving a present state command.

6. A method according to Claim 5, characterized by each sub-unit (2) being structurally and functionally independent of the other sub-units (2).

7. A method according to Claim 5, characterized by each control sub-unit (2) supplying the arbitration block (3) with a predetermined value representing a predetermined state of operation or a neutral state.

8. A method according to Claim 5, characterized by each sub-unit (2) proposing, based on the type of instruction to be executed, a possible state of its own to the arbitration block (3), and by only one in said plurality being in an active state, all the others being in a so-called neutral state.

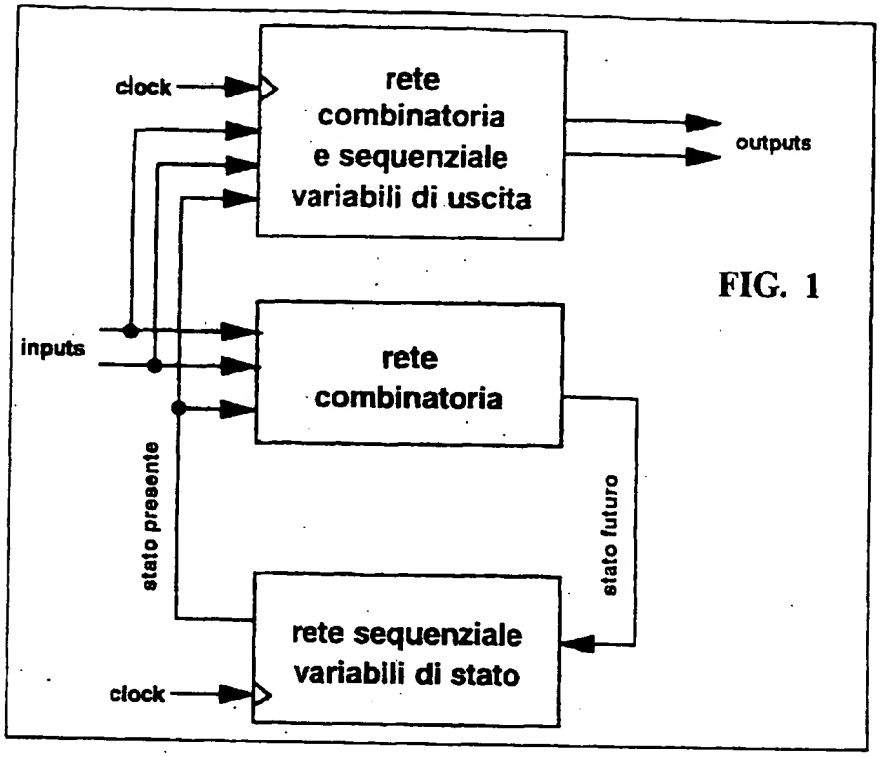

clock
rete combinatoria e sequenziale variabili di uscita
outputs
inputs
rete combinatoria
stato presente
stato futuro
rete sequenziale variabili di stato
clock

FIG. 1

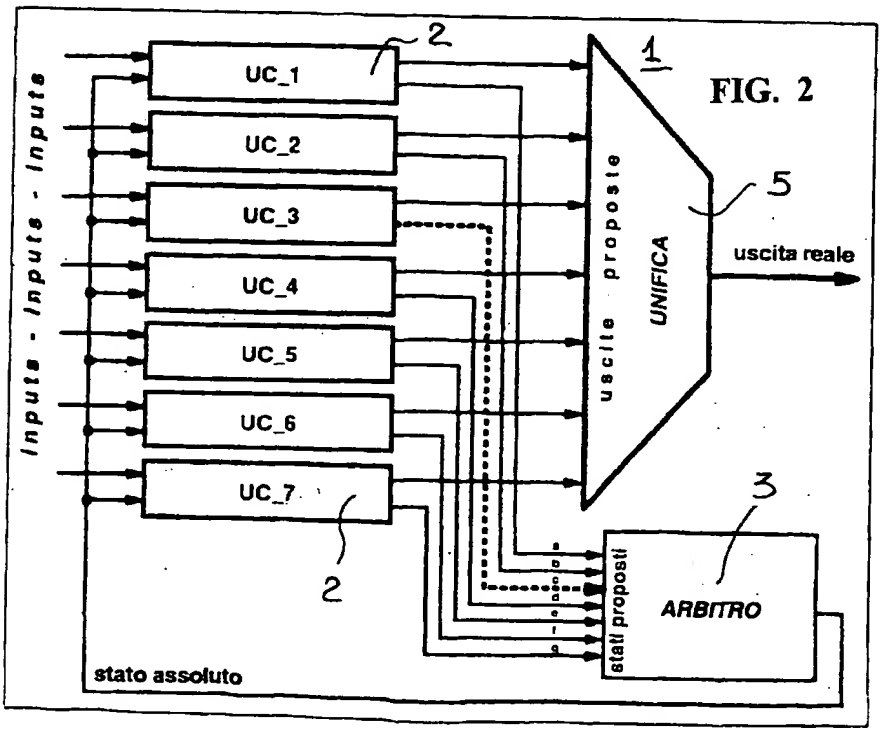

Inputs - Inputs - Inputs
UC_1
UC_2
UC_3
UC_4
UC_5
UC_6
UC_7
2
2
1
uscite proposte
UNIFICA
5
uscita reale
3
ARBITRO
stati proposti
stato assoluto

FIG. 2

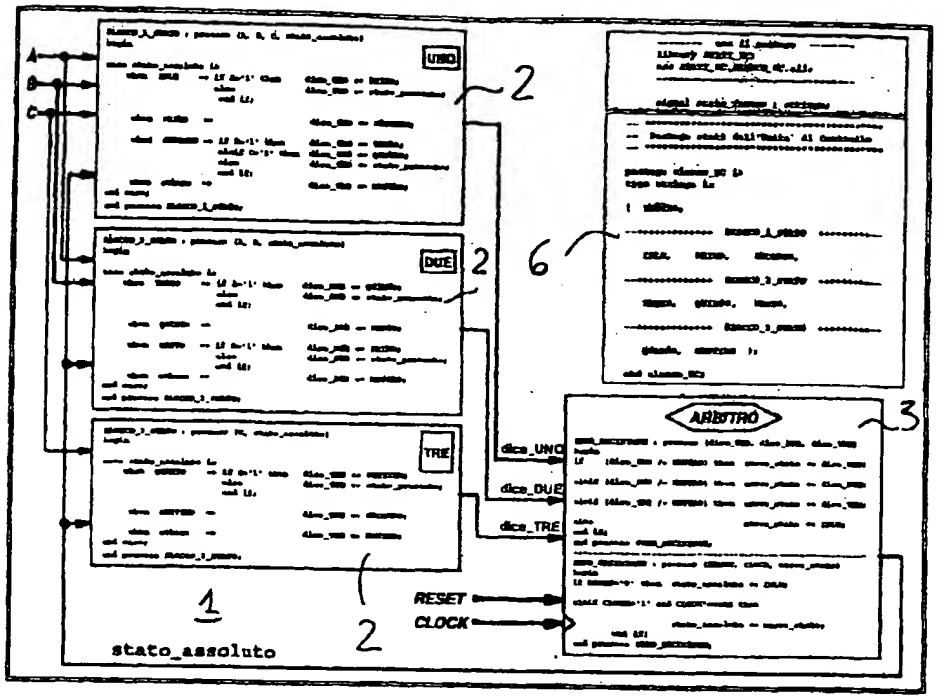
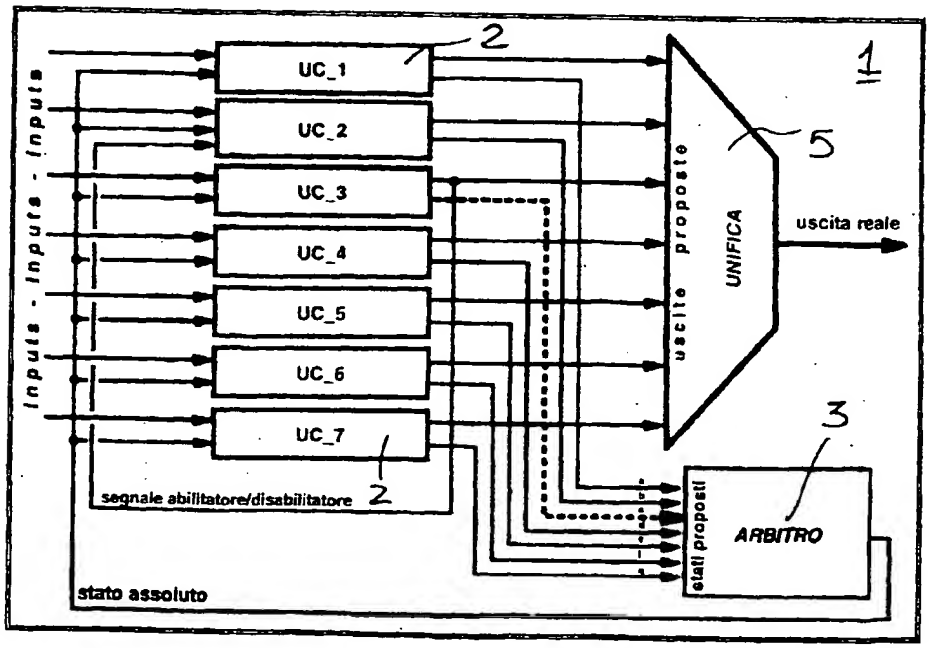

A
B
C
UNO
DUE
TRE
2
2
2
6
ARBITRO
3
dice_UNO
dice_DUE
dice_TRE
RESET
CLOCK
1
2
stato_assoluto

FIG. 3

FIG. 3A


Inputs - Inputs - Inputs
UC_1
UC_2
UC_3
UC_4
UC_5
UC_6
UC_7
2
segnale abilitatore/disabilitatore
2
stato assoluto
uscite proposte
UNIFICA
5
uscita reale
1
3
stati proposti
ARBITRO

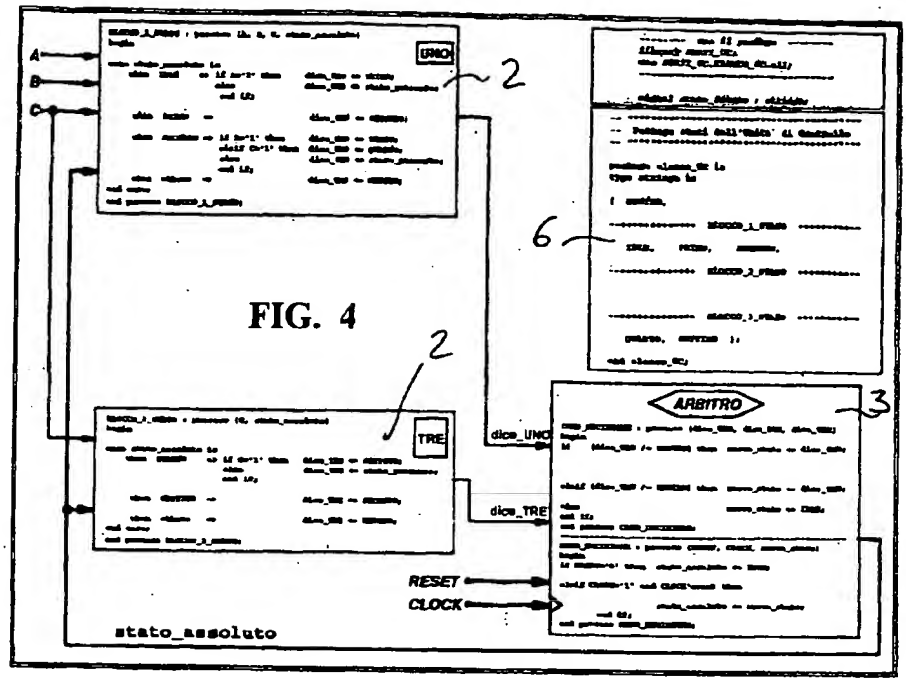

A
B
C
UNO
2
6
2
TRE
ARBITRO
3
dice_UNO
dice_TRE
RESET
CLOCK
stato_assoluto

FIG. 4

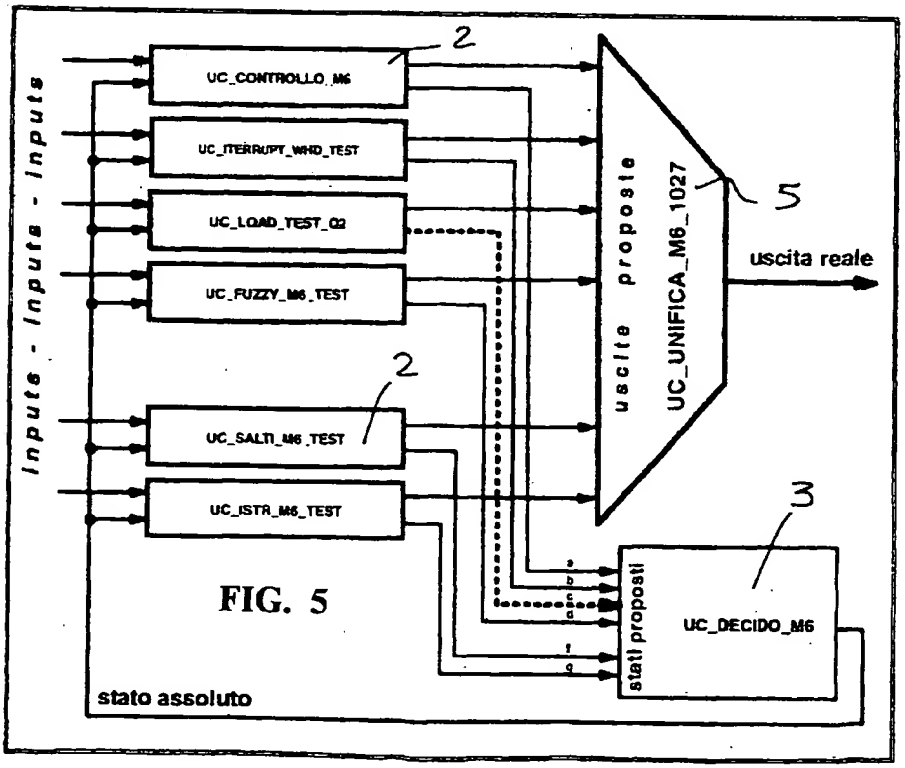

Inputs - Inputs - Inputs
UC_CONTROLLO_M6
UC_ITERRUPT_WHD_TEST
UC_LOAD_TEST_02
UC_FUZZY_M6_TEST
UC_SALTI_M6_TEST
UC_ISTR_M6_TEST
2
2
uscite proposte
UC_UNIFICA_M6_1027
5
uscita reale
3
stati proposti
UC_DECIDO_M6
stato assoluto

FIG. 5

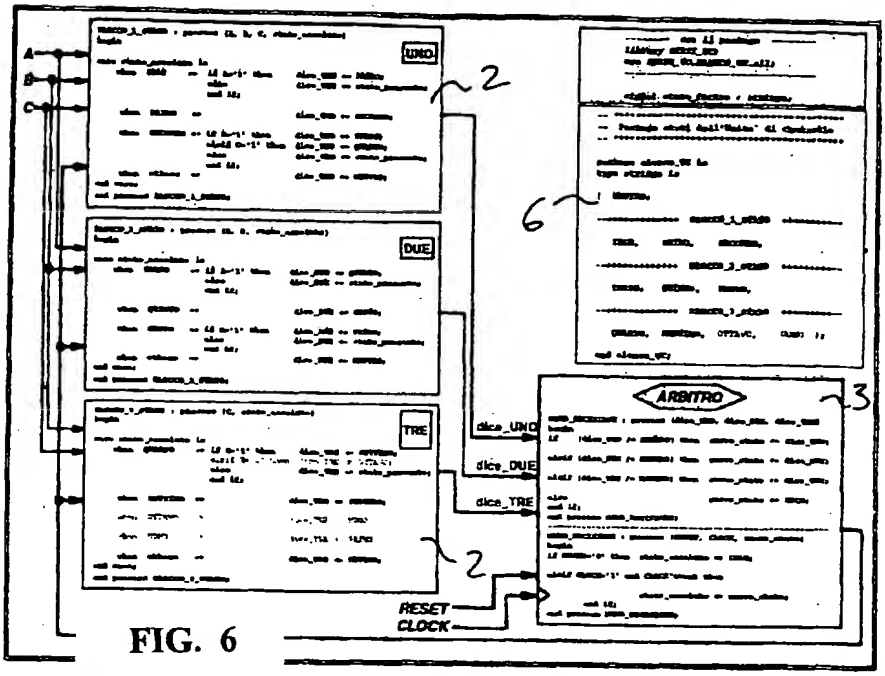

A
B
C
UNO
DUE
TRE
2
6
ARBITRO
3
dice_UNO
dice_DUE
dice_TRE
RESET
CLOCK

FIG. 6

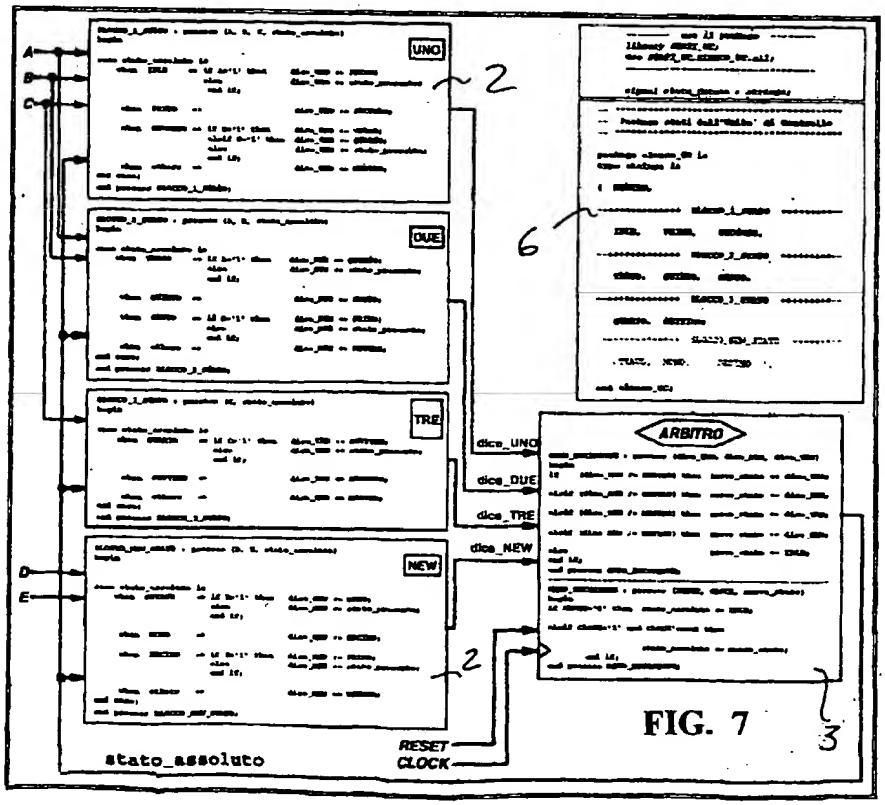

A
B
C
UNO
DUE
TRE
2
6
ARBITRO
dice_UNO
dice_DUE
dice_TRE
dice_NEW
D
E
NEW
2
3
RESET
CLOCK
stato_assoluto

FIG. 7

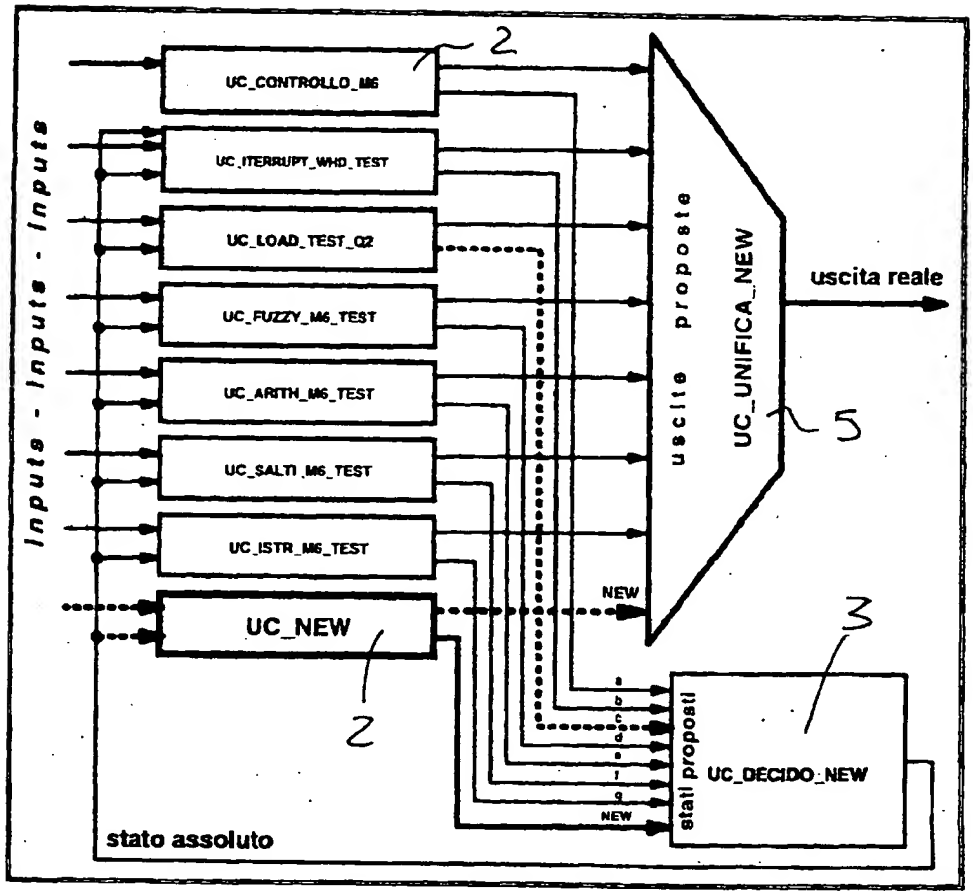

Inputs - Inputs - Inputs
UC_CONTROLLO_M6
UC_ITERRUPT_WHD_TEST
UC_LOAD_TEST_02
UC_FUZZY_M6_TEST
UC_ARITH_M6_TEST
UC_SALTI_M6_TEST
UC_ISTR_M6_TEST
UC_NEW
2
2
uscite proposte
UC_UNIFICA_NEW
5
uscita reale
NEW
3
a
b
c
d
e
f
g
NEW
stati proposti
UC_DECIDO_NEW
stato assoluto

FIG. 8

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 99 83 0375

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	US 3 918 030 A (WALKER RICHARD L) 4 November 1975 (1975-11-04) * column 3, line 60 - column 4, line 20 * * column 4, line 39 - line 45 * * column 5, line 35 - line 50 *	1-8	G06F9/318
X	US 3 760 369 A (KEMP J) 18 September 1973 (1973-09-18) * column 3, line 30 - line 39 * * column 4, line 29 - line 56 * * figure 3 *	1-8	
A	US 4 484 268 A (THOMA NANDOR G ET AL) 20 November 1984 (1984-11-20) * column 1, line 55 - column 2, line 17 *	1	
A	DESAUTELS J C ET AL: "NEW INSTRUCTION AND EXTENDED INSTRUCTION HANDLING" IBM TECHNICAL DISCLOSURE BULLETIN, vol. 21, no. 1, 1 June 1978 (1978-06-01), page 201/202 XP002029309 ISSN: 0018-8689 * the whole document *	1	TECHNICAL FIELDS SEARCHED (Int.Cl.7) G06F
A	MOSER C W: "INCREASING AN INSTRUCTION SET WITHOUT INCREASING WORD LENGTH" PATTERN RECOGNITION, vol. 48, no. 3, 6 February 1975 (1975-02-06), page 114/115 XP002030199 ISSN: 0031-3203	1	

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	21 October 1999	Moraiti, M

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 99 83 0375

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

21-10-1999

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 3918030 A	04-11-1975	NONE	
US 3760369 A	18-09-1973	CA 990411 A	01-06-1976
		DE 2322674 A	13-12-1973
		FR 2195372 A	01-03-1974
		GB 1358534 A	03-07-1974
		IT 981606 B	10-10-1974
		JP 49051839 A	20-05-1974
		JP 53042380 B	10-11-1978
US 4484268 A	20-11-1984	EP 0087008 A	31-08-1983
		JP 1760270 C	20-05-1993
		JP 4035778 B	12-06-1992
		JP 58144262 A	27-08-1983

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82